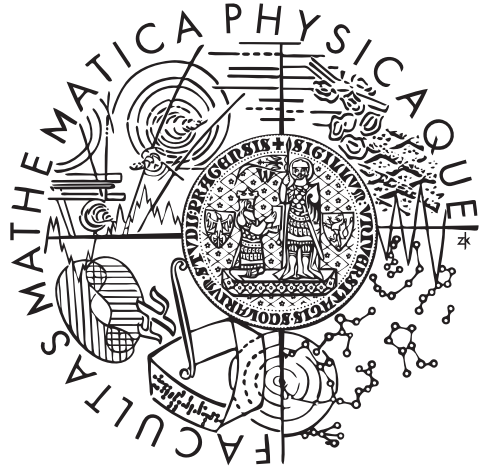
Obecná informatika

Přednášející Putovních přednášek

Matematicko-fyzikální fakulta Univerzity Karlovy v Praze

Podzim 2012



Putovní přednášky

- Obecná informatika
- Lineární algebra
- (Ne)bezpečnost v počítačových sítích
- Hledání cest
- Povídání o studiu nejen na MFF

Cíle přednášek

- Představit vysokoškolskou informatiku
- K čemu je dobré učit se matematiku
- Jak vypadá studium na matfyzu

Přednáška o obecné informatice

- Jaké problémy řešíme?
- Postup řešení – algoritmus
- Porovnávání algoritmů dle rychlosti
- Příklady

Problémy v informatice

- Úloha, kterou lze řešit počítačem
  - Případně provádí řešení robot nebo člověk
- Počítač dostane vstupní data a má něco spočítat
- Například:
  - Vypiš číslo na obrazovku
  - Seřaď čísla dle velikosti
  - Jaká je nejkratší cesta z Aše do Brna?
  - V jakém pořadí má projít pošťák vesnicemi?

Řešení problému

- Hledáme postup, jak úlohu vyřešit
  - Pro jakákoliv vstupní data
- Takový postup nazýváme **algoritmem**
- Počítačový program je algoritmus zapsaný v programovacím jazyce

Příklad algoritmu

- Výpočet faktoriálu čísla n, tedy $n! = n \cdot (n-1) \cdot (n-2) \cdot \ldots \cdot 3 \cdot 2 \cdot 1$
  - Proměnnou *faktorial* nastavíme na 1 a pak násobíme čísly od 1 do n
- Zápis v programovacím jazyce Pascal:

```
var i, n, faktorial: integer;
begin
    read(n);
    faktorial := 1;
    for i := 1 to n do begin
        faktorial := faktorial * i;
    end;
    writeln(faktorial);
end.
```

Správnost algoritmu

- Intuitivně, pokud vydá správný výsledek . . .
- . . . pokud se však zastaví!
- Správnost je třeba zdůvodňovat, často není vidět

Porovnávání algoritmů

- Chceme rychlé algoritmy nenáročné na paměť
- Jak měřit rychlost algoritmů?
  - Stopovat čas běhu na různých vstupních datech
  - Počítat operace, které algoritmus provede

Časová složitost algoritmu

- Závislost počtu provedených operací na velikosti vstupu (N)
  - Lze vyjádřit funkcí, např. $10 \cdot N^2 + 100 \cdot N + 42$
- Zajímají nás velké vstupy
  - Méně významné členy lze vynechat, zbude $10 \cdot N^2$
  - Násobící konstanty také vynecháváme, dostaneme N^2
- Získáme **asymptotickou časovou složitost**, značí se $O(N^2)$
- Náročnost algoritmu na paměť se měří podobně

Časová složitost algoritmu

- Doba běhu algoritmu s časovou složitostí $f(N)$ pro vstupní data velikosti N:
  - Pokud počítač provede 1 milion operací za sekundu

$f(N)$	Velikost dat (N)						
	20	40	60	80	100	500	1000
N	20 μs	40 μs	60 μs	80 μs	0.1 ms	0.5 ms	1 ms
$N \log N$	86 μs	0.2 ms	0.35 ms	0.5 ms	0.7 ms	4.5 ms	10 ms
N^2	0.4 ms	1.6 ms	3.6 ms	6.4 ms	10 ms	0.25 s	1 s
N^3	8 ms	64 ms	0.22 s	0.5 s	1 s	125 s	17 min
2^N	1 s	11.7 dní	36 000 let				
$N!$	77 000 let						

Vyhledávání v seznamu čísel

- Máme dlouhý seznam N čísel, např. 5 1 42 12 7 9 7 2 3
- Chceme zjistit, jestli se v ní nachází číslo X
  - Projdeme posloupnost a každé číslo porovnáme s X
- Co kdyby byla čísla seřazená od nejmenšího po největší?
  - Podíváme se na číslo v půlce seznamu a porovnáme ho s X
  - Je-li např. menší než X, může se X nacházet jen vpravo od půlky
  - Půlíme úsek seznamu, kde může číslo X být
  - Algoritmus se jmenuje **půlení intervalu**
  - Časová složitost $O(\log N)$
- Hodí se tedy umět řadit čísla

Řazení čísel

- Chceme seřadit posloupnost N čísel
- Bublinkový algoritmus:
  - V jednom průchodu polem porovnává sousední prvky od začátku
  - Pokud jsou v opačném pořadí, než mají být, prohodí je
  - Pole projde N-krát
- Jaká je jeho časová složitost?
- Jak zdůvodnit správnost?
- Musíme *vždy* projít pole N-krát?

Rekurzivní algoritmus

- Volá sám sebe – typicky na menší data
  - Rekurzivní funkce v programu volá sama sebe
- Příklad: faktoriál

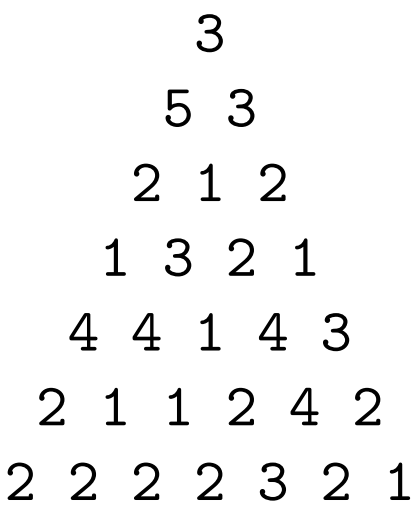
```
function faktorial(n: integer): integer;
begin
    if n = 1 then
        return 1
    else
        return n * faktorial(n - 1);
end;
```

Řazení čísel rekurzivně

- **Třídění sléváním**
- Rozdělíme pole na dvě poloviny
  - Pokud mají víc než 1 prvek
- Obě poloviny setřídíme rekurzivně
- Poloviny pole slijeme do jednoho setříděného pole
  - Odebíráme menší z prvků na začátcích polovin

Úloha s lyžařem

- Lyžař sjíždí sjezdovku a sbírá mince



Shrnutí

- Máte problém?
- Vymyslete algoritmus.
- Zdůvodněte správnost.
- Určete časovou složitost.
- Problém vyřešen :-)

Děkuji za pozornost.
Dotazy?